Sparsity-based optimization of two lifting-based wavelet transforms for semi-regular mesh compression

Aymen Kammoun, Frédéric Payan, Marc Antonini

Sparsity-based optimization of two lifting-based wavelet transforms for semi-regular mesh compression

Aymen Kammoun, Frédéric Payan and Marc Antonini

Laboratoire I3S, UMR CNRS - Université de Nice Sophia Antipolis, France

Abstract

This paper describes how to optimize two popular wavelet transforms for semi-regular meshes, using lifting scheme. The objective is to adapt multiresolution analysis to the input mesh to improve its subsequent coding. Considering either the Butterfly- or the Loop-based lifting schemes, our algorithm finds at each resolution level an optimal prediction operator P such that it minimizes the L_1-norm of the wavelet coefficients. The update operator U is then recomputed in order to take into account the modifications to P. Experimental results show that our algorithm improves on state-of-the-art wavelet coders.

Keywords: Wavelets, Lifting scheme, Semi-regular mesh, Compression, Butterfly, Loop, Sparsity, Optimization.

1. Introduction

Wavelets have their roots in approximation theory [1] and signal processing [2] in the late eighties. Since then, wavelets are the most popular technique for representing data in a multiresolution way. They have been used for a vast number of applications: physic, biomedical signal analysis, image processing, and so on. But wavelets have been particularly designed for data coding, because they guarantees compact representation of transformed data, and consequently high compression performances.

In computer graphics, the compact representation is not the sole attractive feature of wavelets. Indeed, current high-resolution acquisition techniques produce highly detailed and densely sampled surface meshes. Not only these massive monoresolution data are difficult to handle and store, but they are also awkward for fast and progressive transmission in bandwidth-limited applications. Wavelets tackle such issues, the multiresolution structure (Figure 1) making the progressive processing easier.

A problem for applying wavelets on meshes is the irregular sampling (unlike still images or videos). Despite the development of wavelets for irregular meshes [3, 4], a popular solution is to remesh the input mesh semi-regularly (for instance with [5, 6, 7]) before applying wavelets. The principle is to resample the surface geometry while providing a subdivision connectivity. The output is called a semi-regular mesh, and wavelet filtering is finally more efficient.

1.1. Related work

Lounsbery *et al.* are considered as pioneers in the development of wavelets for surface meshes of arbitrary topological type [8]. They proposed a technique to construct wavelets from any local, stationary, continuous, uniformly convergent subdivision schemes such as Catmull-Clark [9], Loop [10], or Butterfly [11]. The subdivision scheme represents the synthesis filter, and the analysis filter is derived from it. Two filters are finally applied on the input mesh during analysis providing respectively a mesh of low resolution (low-pass filtering), and a set of wavelet coefficients (high-pass filtering).

Inspired by the work of Lounsbery *et al.*, and by the work of Donoho concerning interpolating wavelet transforms [12], Schröder and Sweldens presented how building wavelets for scalar functions specifically defined on a sphere [13]. They are not the first constructing wavelets on the sphere. The pioneers are Dahlke *et al.* [14], who used a tensor product basis where one factor is an exponential spline. A continuous transform and its semi-discretization have been also proposed by Freeden and Windheuser [15]. Nevertheless, the work of Schröder and Sweldens in [13] is remarkable because it

Email address: {kammoun,fpayan,am}@i3s.unice.fr (Aymen Kammoun, Frédéric Payan and Marc Antonini)

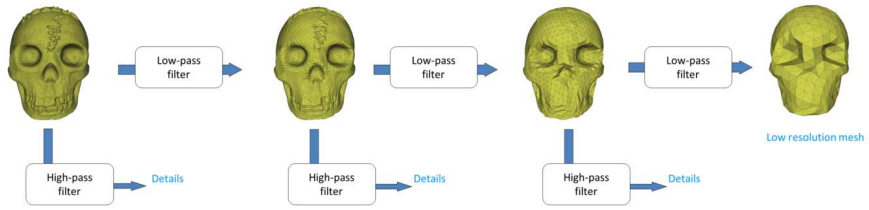


Figure 1: Overview of a wavelet decomposition.

is the first showing how the *lifting scheme* [16] is particularly relevant to construct biorthogonal wavelets with nice properties, and how the resulting wavelet filters are easy to implement (local vertex-manipulating filters). Moreover, this technique is parameterization independent. Kovacevic and Sweldens then generalized the concept of wavelets for any kind of meshes [17]. They showed that the lifting scheme allows to construct filter banks and wavelets for any lattice, any dimension, and any number of primal/dual vanishing moments. They also showed that only two lifting steps are needed (predict and update), but one condition is that the associated scaling functions are interpolating.

Until this work, most of wavelet transforms for semi-regular meshes were based on interpolating subdivision schemes, in particular on the Butterfly scheme. However, a Loop-based wavelet transform was proposed in 2000 by Khodakovsky *et al.* [18]. The approximating Loop subdivision scheme is used during synthesis as low pass reconstruction filter, whereas the associated high-pass filter is derived from it by applying a quadrature mirror construction. The drawback of this approach arises during wavelet analysis, because filters cannot be directly applied. Contrary to wavelet transforms based on lifting scheme, the wavelet coefficients and the low resolution mesh are obtained by solving sparse linear systems depending on the two low- and high-pass reconstruction filters. In 2004, Bertram overcame this problem by proposing a biorthogonal Loop-based wavelet construction based on the lifting scheme [19]. This is also the case of Li *et al.* who proposed in parallel a reversible (but unlifted) Loop-based wavelet transform [20]. Finally, in 2008, Charina and Stöckler proposed to tackle this drawback by using tight wavelet frames [21], which leads to the use of the same scheme during reconstruction and decomposition.

Compression allows compact storage and/or fast transmission in bandwidth-limited applications of massive meshes, and many techniques have been already proposed [22]. To our knowledge, wavelet-based coders that take semi-regular meshes as input are the most efficient, because of their piecewise sampling regularity allowing efficient wavelet decomposition. We briefly present the main works in this domain.

The first wavelet-based coder (often called PGC) for semi-regular meshes was proposed by Khodakovsky *et al.* [18]. This coder is based on multi-scale quadtree structures and supports quality scalability. The authors propose a Loop-based wavelet transform (presented in previous section), but any wavelet transform could be used. A zerotree coder followed by an entropy encoding are applied in parallel on each component (tangential and normal) of the wavelet coefficients computed in a local frame. This coder has been also proposed for *normal meshes* [23]. The only difference is the choice of the wavelet transform. The authors uses the *unlifted* Butterfly-based wavelet transform (*i.e.*, without update step), optimal for this kind of meshes.

Then, several allocation techniques [24, 25, 26, 27] were proposed for improving the coding performances of the wavelet coders. The principle is to use a bit allocation process during the quantization step in order to analytically optimize the rate-distortion tradeoff, in other words, reach the maximal quality for a minimal file size (or *vice versa*).

Recently, a coder providing both resolution and quality scalability was proposed by Denis *et al.* (2010) [28]. This coder exploits the intraband or composite statistical dependencies between the wavelet coefficients. By following an information-theoretic analysis of these statistical dependencies, the wavelet subbands are independently encoded using octree-based coding techniques and a context-based entropy coding. This coder provides better results than PGC, and similar results with [24] that is not quality scalable.

1.2. Motivation and Contributions

One limitation of wavelets for meshes is that the structure is fixed. For instance, many wavelet coders

use the Butterfly-based scheme [13]. From a compression point-of-view, this wavelet is relevant for smooth surfaces because of the interpolating effect of the Butterfly scheme used as predictor, which produces small coefficients. But this scheme is less efficient for other kinds of surfaces, with high frequency variations or salient features, for instance. Finally, a wavelet changing in function of the geometric features of the input mesh could be a relevant tool. When the transforms are lifting-based, this can be finally achieved by *adapting* the predict and update steps [29] to the input mesh.

Therefore we propose an algorithm for optimizing two popular lifting-based wavelet transforms for semi-regular meshes: the Butterfly-based scheme [13], and the Loop-based one [19]. Our motivation is to improve the performances of the state-of-the-art wavelet coders, by adapting the multiresolution analysis tool to the features of the input mesh. The basic idea is to find, for a given semi-regular mesh, the prediction operator that maximizes the sparsity of wavelet coefficients at each level of resolution. Indeed, it is well known in information theory that maximizing the data sparsity improves the coding performances [30].

The idea of adapting the prediction step of the Butterfly-based scheme has been already introduced in [31]. The main contributions of the current paper are:

- More technical details about the optimization algorithm for the Butterfly-based lifting scheme;
- A more robust method for computing the update operator for this scheme. The reason is that the technique proposed in [31] sometimes fails because of a potential null divisor;
- An extension of the optimization algorithm to the Loop-based lifting scheme [19], by taking into account the features of this scheme.

The rest of this paper is organized as follows. Section 2 introduces notions about semi-regular meshes and lifting scheme for semi-regular meshes. Section 3 and 4 present our contributions, respectively for the Butterfly-based and the Loop-based lifting schemes. Section 5 shows some experimental results, and we finally conclude in section 6.

2. Background and notations

2.1. Semi-regular meshes

A semi-regular mesh M^L is based on a mesh hierarchy M^l ($l \in \{0, 1, ..., L\}$) that represents a given surface at different levels of details, or resolutions (see Fig. 2). M^0 corresponds to the lowest resolution, and is called the *base mesh*. M^l is a subdivided version of M^{l-1}, and corresponds to l^{th} resolution. Subdivision consists in splitting each triangle into four smaller ones by adding new vertices on each edge, and then updating their position to fit as closely as possible to the original surface. M^L corresponds to the highest resolution of the given surface.

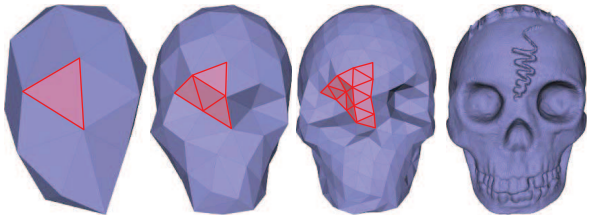


Figure 2: Structure of a semi-regular mesh.

2.2. Wavelet filtering for semi-regular meshes

Let us denote $\mathbf{V}^j$ the set of vertices of a given mesh M^j, defined by their position in the Euclidean space. Applying a wavelet transform to M^j gives one mesh of lower resolution M^{j-1} defined by a set of vertices $\mathbf{V}^{j-1}$, and a set $\mathbf{C}^j$ of 3D wavelet coefficients. Figure 3 illustrates this decomposition.

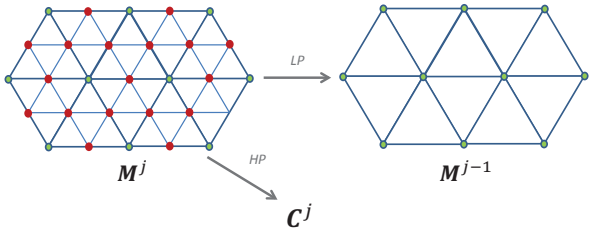


Figure 3: Wavelet analysis of a semi-regular mesh. M^j is transformed into a mesh M^{j-1} of lower resolution and a set $\mathbf{C}^j$ of coefficients (relative to the red dots).

2.3. Lifting scheme

An efficient tool for building wavelet transforms is the lifting scheme [16]. The main feature of the lifting scheme is that all constructions are derived in the spatial domain while the traditional approach relies on the frequency domain. The lifting scheme has several advantages [32], in particular for surface meshes. For example, the lifting scheme i) leads to algorithms that can be generalized to complex geometric situations; ii) allows in place transformation, reducing the necessary amount of memory; iii) leads to a reversible

implementation (analysis/synthesis), faster than implementation based on filter banks. Nevertheless, the lifting scheme has few drawbacks. For instance, the stability of the transform is not guaranteed by construction.

The canonical lifting scheme (Figure 4) is based on three steps:

- **Split**. We split the original data set into two subsets. In our context, the vertices $\mathbf{V}^j$ are split into two subsets $\mathbf{V}_0^j$ and $\mathbf{V}_1^j$ (green and red dots on the figure 3);
- **Predict**. We take as input the subset $\mathbf{V}_0^j$ and predict the positions of the vertices $\mathbf{V}_1^j$ by using an operator P, so that $\mathbf{V}_1^j = P(\mathbf{V}_0^j)$. The prediction errors are the wavelet coefficients $\mathbf{C}^j$;
- **Update**. We take as input $\mathbf{C}^j$ and modify the positions of the vertices $\mathbf{V}_0^j$ by using an operator U and a gain ($\times 2$). We finally obtain the low resolution mesh M^{j-1}, defined by a set of vertices $\mathbf{V}^{j-1}$.

For the synthesis, we only have to reverse the order and the sign of the different steps (Figure 5).

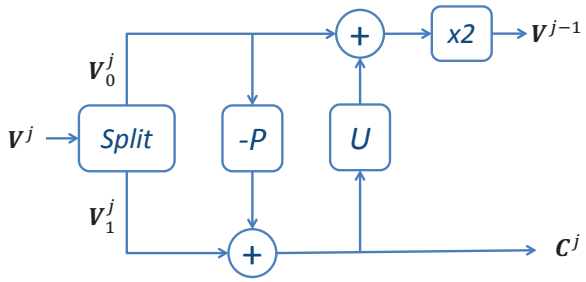


Figure 4: Analysis lifting scheme for semi-regular meshes.

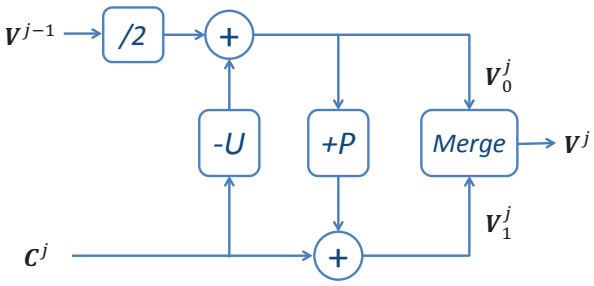


Figure 5: Synthesis lifting scheme for semi-regular meshes.

3. Adaptive Butterfly-based lifting scheme

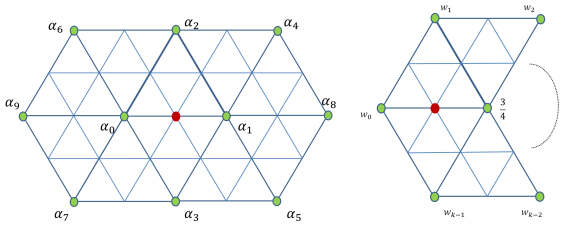
In this section we present our optimization algorithm for adapting the Butterfly-based lifting scheme to the input mesh. This algorithm consists in finding for each resolution the optimal operators P such as the sparsity of the wavelet coefficients is maximized [31]. This algorithm produces new stencils for each level of resolution.

Our wavelet coefficients are vector valued (three coordinates x, y, and z) and represented in local frames depending on the surface normals [18]. When using local frames, the state of the art wavelet coders consider the tangential components (x and y), and the normal components (z) separately. We thus propose to compute two prediction operators, P_{xy} and P_z, according to the two sets of components. The algorithm presented below will be used to get P_{xy} and in parallel P_z. For the convenience of the readers, we define hereinafter P for both P_{xy} and P_z.

3.1. Optimization of the operator P

The predict step is based on the *modified Butterfly scheme* [33] defined by two stencils: the regular and the irregular (Figure 6).

- For a given vertex $\mathbf{v} \in \mathbf{V}_1^j$, the regular stencil is used when its direct neighbors belonging to $\mathbf{V}_0^j$ are regular (*i.e.*, valence 6).
- The irregular stencil is chosen for any other case.



(a) Regular stencil. (b) Irregular stencil.

Figure 6: Prediction stencils of the Butterfly-based lifting scheme.

Our algorithm optimizes only the regular stencil, because there are too many configurations for the irregular one (depending on the valence of the neighbors). Optimizing all the potential irregular stencils may produce a critical amount of side information (necessary for reconstructing the meshes during synthesis).

Considering the level of resolution j, the set $\mathbf{C}^j$ of wavelet coefficients is computed by using

$$\mathbf{C}^j = \mathbf{V}_1^j - P(\mathbf{V}_0^j). \quad (1)$$

In order to increase the coding performances, we want $\mathbf{C}^j$ to become as sparse as possible. One solution proposed in the literature is to minimize the L_1-norm of this set [34]. So, maximizing the sparsity can be seen as the minimization problem

$$\min_{\boldsymbol{\alpha}^j} \|\mathbf{V}_1^j - P_{\boldsymbol{\alpha}^j}(\mathbf{V}_0^j)\|_1, \tag{2}$$

where $\boldsymbol{\alpha}^j$ defines the weights of P for the level of resolution j.

To solve this problem, we define the unknown vector $\mathbf{x} = \left(\alpha_0^j, \alpha_1^j, \ldots, \alpha_9^j\right)^T$ containing the ten weights of P, and $\mathcal{N}(\mathbf{v})$ the set of ten neighbor vertices of $\mathbf{V_0^j}$ and depending on the regular stencil. We now define the matrix A of dimension $(n_r \times 10)$

$$A = \left[\mathcal{N}(\mathbf{V}_1^j(0))\,;\ \mathcal{N}(\mathbf{V}_1^j(1))\,;\ \ldots \mathcal{N}(\mathbf{V}_1^j(n_r-1))\right], \tag{3}$$

where n_r is the number of vertices of $\mathbf{V}_1^j$ on which the regular stencil is applied at this resolution. We then define the vector $\mathbf{b}$

$$\mathbf{b} = \left(\mathbf{V}_1^j(0), \mathbf{V}_1^j(1), \ldots \mathbf{V}_1^j(n_r-1)\right)^T. \tag{4}$$

Finally, (2) can be solved by minimizing one function f defined by

$$\begin{array}{rccl} f: & \mathbb{R}^{10} & \longmapsto & \mathbb{R} \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) = \|A\mathbf{x} - \mathbf{b}\|_1. \end{array} \tag{5}$$

Since the Butterfly scheme is symmetric, we can write

$$\begin{array}{rcl} \alpha_0 & = & \alpha_1 \\ \alpha_2 & = & \alpha_3 \\ \alpha_4 & = & \alpha_5 = \alpha_6 = \alpha_7 \\ \alpha_8 & = & \alpha_9. \end{array} \tag{6}$$

Finally, the function f has only four unknown values (α_0, α_2, α_4 and α_8) and can be written as following

$$\begin{array}{rccl} f: & \mathbb{R}^{4} & \longmapsto & \mathbb{R} \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) = \|A\mathbf{x} - \mathbf{b}\|_1. \end{array} \tag{7}$$

This function f is convex (see Appendix A for details) and bounded by 0, so f has a unique and global minimum. To find this minimum, we use the Nelder-Mead simplex algorithm [35], but other algorithms may be used. The proposed optimization algorithm is applied to each level of resolution, successively from the highest to the lowest.

3.2. Computation of the new operator U

Considering the same level of resolution j and the canonical lifting scheme (Figure 4), the set of vertices $\mathbf{V}^{j-1}$ is obtained by using

$$\mathbf{V}^{j-1} = 2 \times (\mathbf{V}_0^j + U(\mathbf{C}^j)), \tag{8}$$

where U is the update operator depending on a weight γ and associated to the stencil given by Figure 7.

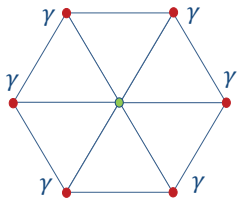


Figure 7: Update stencil of the Butterfly-based lifting scheme.

When the prediction P is based on a subdivision operator, the update U, that depends on P, has to be chosen for obtaining vanishing moments [32]. P being modified at each level of resolution by the optimization algorithm, the weight γ of the update operator U has to be recomputed. In this work, we choose to compute γ such as to preserve the average between $\mathbf{V}^j$ and $\mathbf{V}^{j-1}$. Contrary to [31], we prefer using the robust method proposed in [32]. The principle is to put all the vertices of $\mathbf{V}^{j-1}$ and all the coefficients of $\mathbf{C}^j$ to zero, except one coefficient of $\mathbf{C}^j$ put to 1 (see Figure 8).

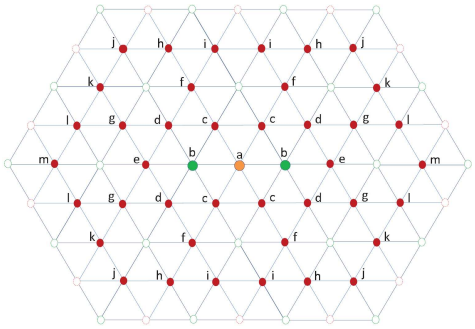


Figure 8: Method for computing γ. The orange dot (a) represents the only non null coefficient; the green dots (b) represent the two sole vertices of $\mathbf{V^{j-1}}$ with non null values after synthesis; the red dots represent the 43 vertices of $\mathbf{V^j}$ with non null values after synthesis.

We apply the synthesis filters, and obtain $\mathbf{V}^j$. A majority of the resulting vertices has null coordinates, except 43 over which the non null coefficient $\mathbf{C}^j$ spread. These vertices are also shown on Figure 8. The value associated to each vertex are given by Table 1.

index	count	value
a	1	$1 - 2\gamma\alpha_0$
b	2	$-\gamma$
c	4	$-\gamma(\alpha_0 + \alpha_2)$
d	4	$-\gamma(\alpha_0 + \alpha_4)$
e	2	$-\gamma(\alpha_0 + \alpha_8)$
f	4	$-\gamma(\alpha_2 + \alpha_4)$
g	4	$-\gamma\alpha_2$
h	4	$-\gamma\alpha_4$
i	4	$-\gamma(\alpha_4 + \alpha_8)$
j	4	$-\gamma\alpha_8$
k	4	$-\gamma\alpha_4$
l	4	$-\gamma\alpha_4$
m	2	$-\gamma\alpha_8$

Table 1: Values of the 43 non null vertices of $\mathbf{V}^j$ obtaining after synthesis from only one non-null coefficient of $\mathbf{C}^j$.

In this case, the average of $\mathbf{V}^j$ is equal to

$$\frac{1}{43}(1 - 2\gamma - 12\gamma\alpha_0 - 12\gamma\alpha_2 - 24\gamma\alpha_4 - 12\gamma\alpha_8) \quad (9)$$

Since the average of $\mathbf{V}^{j-1}$ is null, equation (9) has to be also null. We finally obtain

$$\gamma = \frac{1}{2 + 12\alpha_0 + 12\alpha_2 + 24\alpha_4 + 12\alpha_8}. \quad (10)$$

3.3. Validation of the algorithm

To verify the efficiency of our optimization algorithm, we compare the L_1-norm of the wavelet coefficients obtained with the state of the art Butterfly-based lifting scheme (*Classical*), and with the proposed adaptive scheme (*Optimized I*). The results for Vase Lion are presented in Table 2.

	Classical		Optimized (I)		Optimized (II)	
res	*TC*	*NC*	*TC*	*NC*	*TC*	*NC*
1	83.76	132.21	84.41	126.40	83.80	125.18
2	62.32	134.32	62.21	125.53	62.10	125.50
3	45.80	126.92	45.10	116.87	45.08	116.88
4	31.53	77.88	29.94	69.53	29.93	69.51
5	21.08	42.15	20.22	39.22	20.22	39.21
6	13.81	26.10	13.50	25.78	13.50	25.78

Table 2: L_1-norm of the tangential and normal components (*TC* and *NC*) of the wavelet coefficients obtained with the state of the art Butterfly-based lifting scheme (*Classical*) and with our adaptive scheme (*Optimized (I)*) for Vase Lion. *Optimized (II)* is our adaptive scheme without constraint of symmetry. *res* is the level of resolution.

We observe that the L_1-norm of each subset is globally lower with our adaptive scheme. Similar results are obtained for all the experimented models, proving that our algorithm works well. Nevertheless, for few models (*e.g.* Vase Lion, Table 2), our algorithm does not decrease the L_1-norm of the tangential components of lowest resolution. This issue sometimes arises, only at this resolution, when a majority of vertices are irregular and requires the irregular stencil. In this specific case, our optimization algorithm may not be effective at this resolution but remains efficient for the other resolutions.

We also studied the influence of the constraint of symmetry (6) required by the prediction stencil. The column *Optimized (II)* presents the results of our algorithm without this constraint. We observe that the results are similar. The constrained optimization process being three times faster (see Table 3), we finally retain this variant for the experimentations (Section 5).

4. Adaptive Loop-based lifting scheme

As introduced in Section 1, we now expand our optimization technique to the Loop-based lifting scheme [19]. This transform, illustrated by figure 9, differs from the canonical lifting scheme.

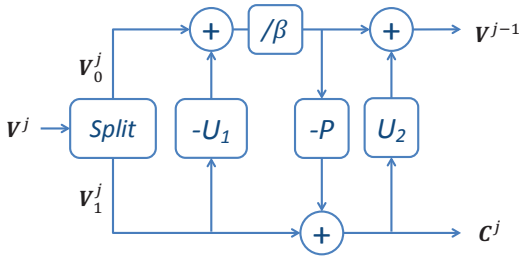


Figure 9: Analysis Loop-based lifting scheme[19].

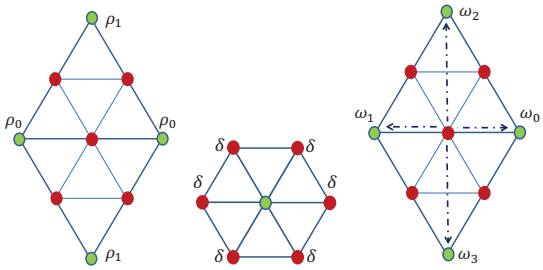
The main features of this scheme are:

- a prediction operator P depending on two weights ρ_0 and ρ_1 (see Figure 10(a));
- an update operator U_1 applied before the predict step depending on a weight δ (see Figure 10(b)). A gain $\frac{1}{\beta}$ (that depends on the valence of the vertices) is also applied;
- a second operator U_2 depending on weights ω (see Figure 10(c)). This step is applied to obtain a biorthogonal wavelet transform.

Note that the two update filters depend on the weights $\boldsymbol{\rho}$ of P.

Model	# resolution	# vertices	Optimized (I)	Optimized (II)
Bimba	7	999426	79	207
Rabbit	6	163842	13	34
Vase Lion	6	675842	55	141

Table 3: Computation time (in seconds) for optimizing the Butterfly-based scheme (MATLAB implementation).



(a) Prediction. (b) First update. (c) Second update.

Figure 10: Stencils of the Loop-based lifting scheme.

4.1. Optimization of the operators P and U_1

We still minimize the L_1-norm of the wavelet coefficients at each level of resolution, but the tangential and the normal components are not considered separately, unlike the Butterfly case. Indeed, if two prediction operators P_{xy} and P_z are computed (for the tangential and the normal components), we obtain two sets of wavelet functions for each vertex, and the orthogonalization done by the update filter U_2 becomes impossible [19]. Thus, at each level of resolution, the algorithm takes the set of vector valued coefficients as input, and gives only one optimized P.

The predictor depends on two parameters ρ_0 and ρ_1. So, the minimization problem for this scheme is $\min_{\{\rho_0,\rho_1\}} \|\mathbf{C}^j\|_1$. But, contrary to the Butterfly-based scheme, the coefficients are obtained after two steps, U_1 and P. The prediction operator P and the first update operator U_1 have to be optimized simultaneously, and the minimization problem becomes

$$\min_{\{\rho_0,\rho_1\}} \|\mathbf{V}_1^j - P\left(\frac{\mathbf{V}_0^j - U_1\left(\mathbf{V}_1^j\right)}{\beta}\right)\|_1. \tag{11}$$

When minimizing this function, the weight δ of U_1 and β must be recomputed at each iteration, since they depend on ρ_0 and ρ_1. For a vertex $\mathbf{v}$ of $\mathbf{V}_0^j$, they are formulated by

$$\beta(n) = \frac{1}{1-\rho_0}\left(\rho_0 + 2\rho_1 \cos\frac{2\pi}{n}\right)^2, \tag{12}$$

$$\delta(n) = \frac{1-\beta(n)}{n}. \tag{13}$$

where n is the valence of $\mathbf{v}$ at this resolution.

To find the optimal weights $\boldsymbol{\rho}^*$ at each resolution, we use the same algorithm than for the Butterfly-based scheme, *i.e.*, the Nelder-Mead simplex algorithm [35]. To respect the constraint of the Loop subdivision scheme, we assume that the sum of the four weights of the prediction stencil is equal to one.

4.2. Computation of the new operator U_2

Now we have to modify the second update operator U_2 that depends on $\boldsymbol{\omega}$. In [19] the author details how computing $\boldsymbol{\omega}$ in function of the weights $\boldsymbol{\rho}$, to finally get a biorthogonal wavelet scheme. Since our algorithm modifies the predictor at each resolution level, we also have to compute new weights $\boldsymbol{\omega}$ at each resolution level.

As explained in [19], we have to solve $A\boldsymbol{\omega} = b$, where A is a symmetric 4×4 matrix

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix},$$

and b is a 4×1 matrix defined by $b = [b_0\ b_1\ b_2\ b_3]^T$. Finally, the key problem is to compute a_{ij} and b_i that depend on $\boldsymbol{\rho}$. Once these weights computed, the weights $\boldsymbol{\omega}$ relative to each vertex of each resolution are obtained by computing $\boldsymbol{\omega} = A^{-1}b$. For the convenience of the readers, details of how computing the terms a_{ij} and b_i can be found in Appendix B.

4.3. Validation of the method

To validate our optimization algorithm for the Loop-based lifting scheme, we compared the L_1-norm of the wavelet coefficients obtained with the classical scheme (*Classical*), and with our adaptive scheme (*Optimized*), for several models. The results for JOAN OF ARC are presented in Table 4. We observe that our optimization algorithm is also efficient with the Loop-based lifting scheme, since the L_1-norm values are always significantly lower with our adaptive scheme.

	Classical		Optimized	
res	*TC*	*NC*	*TC*	*NC*
1	13.63	15.51	8.05	11.65
2	16.49	17.24	13.18	14.96
3	16.02	16.85	13.78	15.37
4	15.89	16.91	14.74	16.06
5	12.70	13.82	12.56	13.80

Table 4: L_1-norm of the tangential and normal components (*TC* and *NC*) of the wavelet coefficients obtained with the state of the art Loop-based lifting scheme (*Classical*) and with our adaptive scheme (*Optimized*) for Joan of Arc. *res* is the level of resolution.

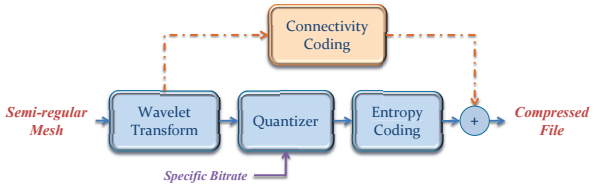


Figure 11: Overview of wavelet coder for semi-regular meshes.

5. Application to the geometry compression

To evaluate if our technique improves the coding of the semi-regular meshes, we now include our adaptive transform into a wavelet coder.

The main steps of a wavelet coder (Figure 11) are:

1. **Wavelet transform**. This stage provides the subbands of 3D wavelet coefficients and the lowest resolution mesh.
2. **Connectivity coder**. The connectivity of the lowest resolution mesh is lossless encoded (for instance, with [36]).
3. **Quantizer.** The subbands of coefficients are quantized so that the minimum reconstruction error is introduced, according to a specific bitrate.
4. **Entropy coding.** The quantized data are entropy coded.

We experiment two coders. The first one is called PGC [18]. This is the first wavelet coder for semi-regular meshes (see section 1.1), and the most popular. The second one is the coder of [24, 25], called hereinafter EDQ. This coder includes a bit allocation that minimizes the reconstruction error by computing the optimal quantization steps $\{q_i\}$ of each subband for a specific bitrate R_{target}. This allocation is done by minimizing the Lagrangian criterion

$$J_\lambda(\{q_i\}) = D_T(\{q_i\}) + \lambda(R_T(\{q_i\}) - R_{target}), \qquad (14)$$

where D_T is the mean square error between the quantized and the original vertices, and R_T is the bitrate. Assuming that the distribution of the tangential and the normal components can be modeled by a Generalized Gaussian Distribution, theoretical models are used for estimating the bitrate and the distortion of each subband. We choose to experiment this coder because it is model-based. We would like to verify if our optimization algorithm is also relevant for such a coder, in other words if our algorithm does not drastically change the statistical properties of the wavelet coefficients.

5.1. Results for the Butterfly-based scheme

Tables 5 to 10 show the values of PSNR and RMS for three models (Vase Lion, Bimba and Rabbit), analyzed with the classical or with our adaptive Butterfly-based scheme, and then compressed with PGC or with EDQ.

The RMS corresponds to the surface-based Root Mean Square error, computed between the original irregular mesh and the reconstructed semi-regular with MESH tools [37]. The PSNR is given by

$$PSNR(dB) = 20log_{10}\frac{BB}{RMS}, \qquad (15)$$

with BB the length of the bounding box diagonal. The bitrates are given in bits per irregular vertex (b/iv). For the adaptive scheme, the bitrate includes the set of optimized weights, needed for decoding.

	Classical		Optimized (I)		**Gain**
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
2.50	5.49	68.03	5.19	68.51	**5.37%**
3.50	4.08	70.60	3.82	71.16	**6.29%**
5.50	2.70	74.20	2.57	74.62	**4.75%**
9.00	1.84	77.52	1.78	77.79	**3.00%**
11.00	1.58	78.86	1.53	79.14	**3.13%**
13.00	1.46	79.54	1.40	79.91	**4.22%**

Table 5: Coding results for Vase Lion when using the Butterfly-based schemes and PGC. The RMS is in multiples of 10^{-4}.

	Classical		Optimized (I)		**Gain**
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
1.75	10.21	81.91	10.14	82.38	**5.28%**
2.50	8.43	85.04	8.24	85.24	**2.20%**
3.50	6.36	87.79	6.31	87.56	**0.77%**
5.50	4.76	90.01	4.65	90.21	**2.31%**
9.00	3.76	92.06	3.70	92.18	**1.42%**
11.00	3.46	92.78	3.43	92.85	**0.87%**

Table 6: Coding results for Bimba when using Butterfly-based schemes and PGC. The RMS is in multiples of 10^{-5}.

	Classical		Optimized (I)		**Gain**
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
1.75	7.81	81.53	7.66	81.70	**2.01%**
2.50	6.41	83.25	6.38	83.29	**0.43%**
3.50	5.69	84.29	5.59	84.43	**1.65%**
5.50	5.04	85.33	4.99	85.43	**1.06%**
9.00	4.61	86.11	4.59	86.15	**0.46%**
13.00	4.48	86.35	4.48	86.36	**0.14%**

Table 7: Coding results for RABBIT when using Butterfly-based schemes and PGC.The RMS is in multiples of 10^{-6}.

	Classical		Optimized (I)		**Gain**
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
2.58	5.06	68.73	4.83	69.14	**4.68%**
3.51	3.87	71.07	3.78	71.27	**2.33%**
5.33	2.78	73.92	2.71	74.17	**2.77%**
8.90	1.78	77.79	1.71	78.15	**4.01%**
10.78	1.62	78.63	1.53	79.11	**5.34%**
12.70	1.45	79.56	1.40	79.88	**3.60%**

Table 8: Coding results for VASE LION when using Butterfly-based schemes and EDQ. The RMS is in multiples of 10^{-4}.

	Classical		Optimized (I)		**Gain**
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
1.55	10.39	80.70	10.29	81.36	**7.33%**
2.17	9.86	83.68	9.21	84.27	**6.58%**
3.03	7.03	86.62	6.87	86.82	**2.23%**
4.75	5.15	89.32	5.02	89.54	**2.42%**
7.85	3.94	91.64	3.90	91.74	**1.09%**
9.60	3.62	92.38	3.55	92.54	**1.90%**

Table 9: Coding results for BIMBA when using Butterfly-based schemes and EDQ. The RMS is in multiples of 10^{-5}.

	Classical		Optimized (I)		**Gain**
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
1.07	9.96	79.42	9.79	79.573	**1.72%**
1.89	7.12	82.34	7.03	82.450	**1.23%**
2.63	6.45	83.19	6.39	83.277	**1.00%**
3.65	5.46	84.64	5.44	84.670	**0.40%**
10.20	4.51	86.30	4.50	86.25	**0.29%**

Table 10: Coding results for RABBIT when using Butterfly-based schemes and EDQ. The RMS is in multiples of 10^{-6}.

Globally the proposed adaptive scheme improves the coding whatever the coder. The maximum gain is 6.29% for VASE LION, 7.33% for BIMBA, and 2.01% for RABBIT. The most significant gains are obtained for VASE LION and BIMBA, in particular at low and medium bitrates, where there is a room for improvement. The gain is lower for RABBIT, even negligible at some bitrates. These results confirm our assumptions: the more the surface has important high frequency variations, the more our adaptive transform is efficient.

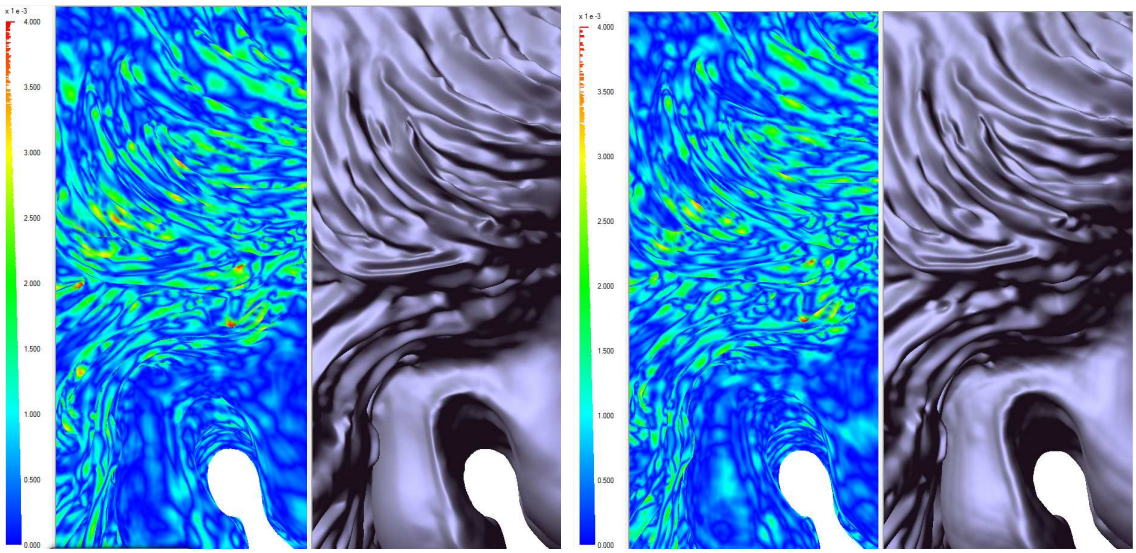
In addition, Figure 12 show the visual benefits of the proposed Butterfly-based technique. This figure shows the distribution of the reconstruction error of VASE LION, analyzed with the classical transform and with the adaptive transform (compressed with EDQ). One can argue that our adaptive wavelet transform tends to better preserve the high frequency details of the non-smooth regions, such as the mane. On the other hand, the classical transform seems to introduce less errors on smooth regions, such as the muzzle. These results confirm the fact that our adaptive transform is better suited for non-smooth regions. Thus, it could be interesting in the future to combine the two transforms in a region-based wavelet coding scheme. For instance, after using a segmentation technique sharing the mesh in two regions (smooth *vs* detailed), we could analyze them separately (smooth region = classical transform, detailed region = adaptive transform). The coding gain should be superior and the visual quality better.

We also compare the efficiency of the two transforms applied on meshes with salient features, for which the classical Butterfly-based scheme is generally less efficient. Figure 13 shows that our adaptive scheme tends to further preserve the salient features of KNOT.

5.2. Results for the Loop-based scheme

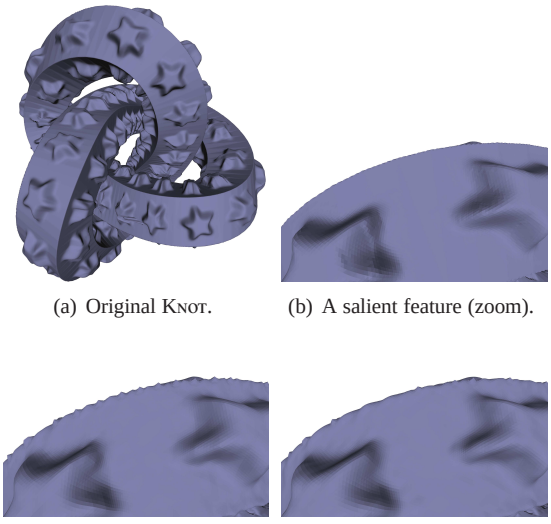
Tables 11 to 13 gives the PSNR and RMS for VASE LION, BIMBA and RABBIT analyzed with the classical or with the adaptive Loop-based scheme, and then compressed with PGC. Our optimization algorithm also improves the coding performances when the Loop-based is used. Globally the gain is even higher than with Butterfly. The maximum gain is 19.55% for VASE LION, 19.73% for BIMBA, and 6.31% for RABBIT, still at low bitrates.

One drawback of the classical Loop-based wavelet is to create distorted low resolution meshes, in particular when the input is complex. By comparing the resolutions of several models, we observe that the proposed adaptive scheme creates low resolution meshes less distorted than with the original scheme. Figure 14 illustrates this fact. This is a very interesting feature, which



(a) Classical wavelet. (b) Adaptive wavelet.

Figure 12: Distribution of the reconstruction error of VASE LION, analyzed with the classical and with the adaptive Butterfly-based scheme (compressed with EDQ). The color corresponds to the magnitude of the distance point-surface computed between the original mesh and the compressed (MESH tools).



(a) Original KNOT. (b) A salient feature (zoom).

(c) Classical wavelet transform. (d) Adaptive wavelet transform.

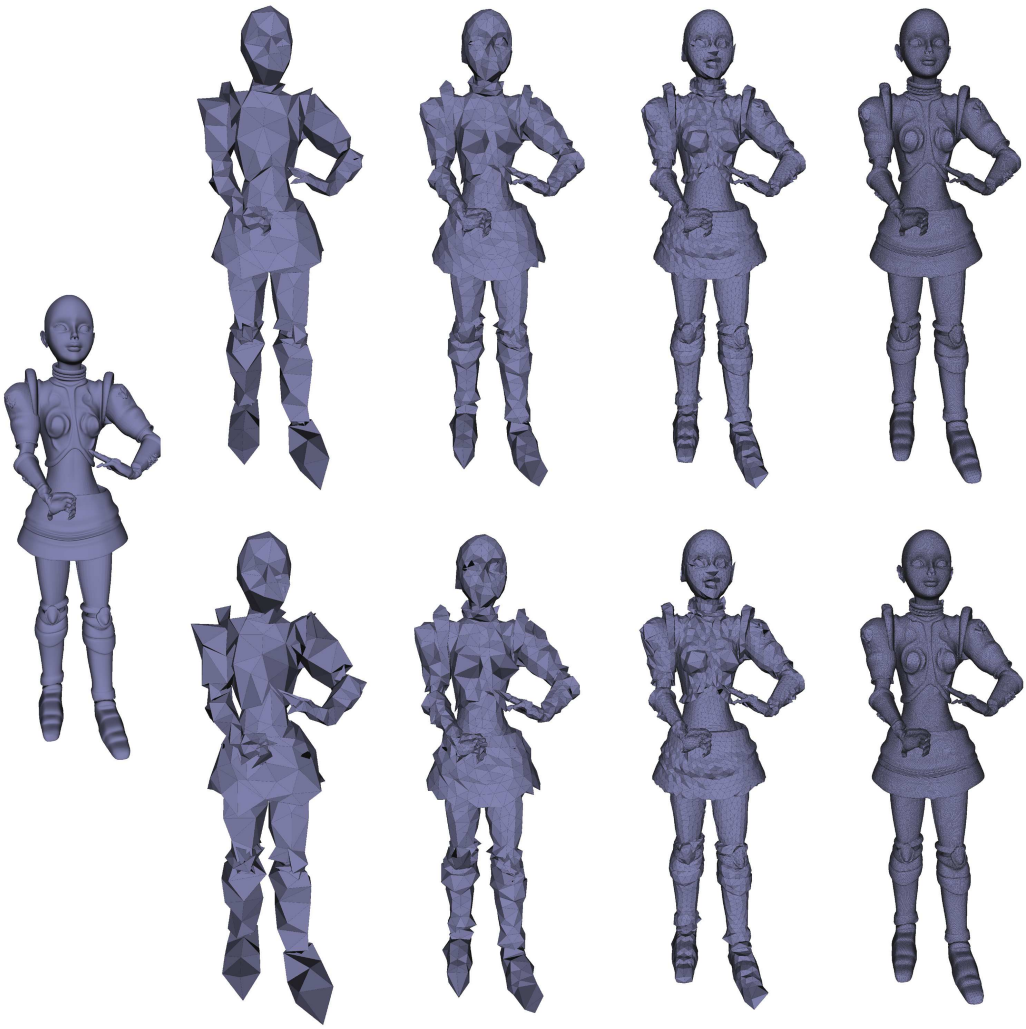
Figure 13: Salient features of compressed models tend to be better preserved with our adaptive Butterfly-based scheme.

	Classical		Adaptive		Gain
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
2.5	10.53	59.13	10.39	59.95	**9.01%**
5.5	6.99	65.92	5.63	67.81	**19.55%**
9	4.47	69.80	4.43	69.89	**1.08%**
11	3.35	72.30	3.26	72.54	**2.70%**

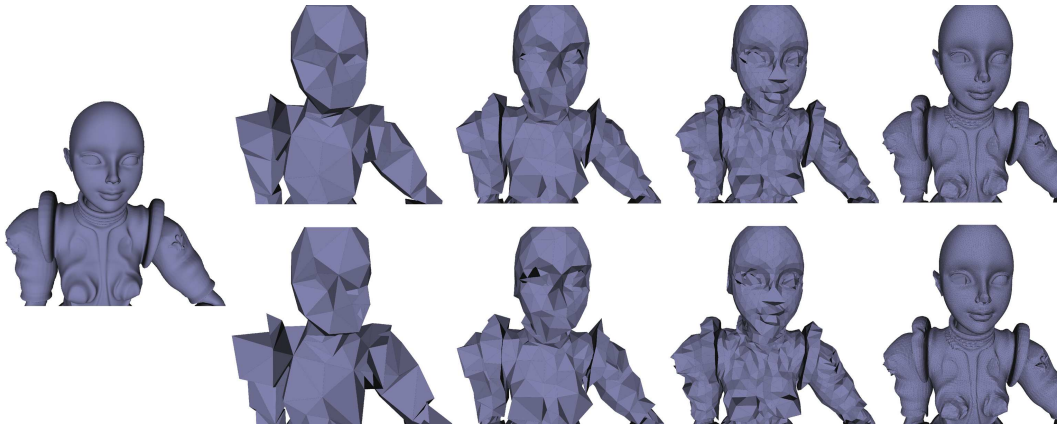
Table 11: Coding results for VASE LION when using Loop-based schemes and PGC. The RMS is in multiples of 10^{-4}.

	Classical		Adaptive		Gain
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
1	4.96	69.64	3.98	71.55	**19.73%**
1.75	2.71	74.90	2.39	76.00	**11.86%**
3.5	1.44	80.36	1.41	80.55	**2.17%**
9	0.78	85.76	0.77	85.86	**1.16%**

Table 12: Coding results for BIMBA when using Loop-based schemes and PGC. The RMS is in multiples of 10^{-4}.



(a) Left: original model; top: our adaptive scheme; bottom: the classical scheme.



(b) Left: original model; top: our adaptive scheme; bottom: the classical scheme.

Figure 14: At low resolutions, the shape of the original model is better preserved with our adaptive Loop-based scheme.

	Classical		Adaptive		Gain
Bitrate	RMS	PSNR	RMS	PSNR	**RMS**
1.75	10.01	79.34	9.54	79.80	**5.18%**
5.5	5.50	84.58	5.47	84.63	**0.63%**
9	4.79	85.79	4.79	85.79	**0.38%**
11	4.60	86.13	4.59	86.16	**0.33%**

Table 13: Coding results for RABBIT when using Loop-based schemes and PGC. The RMS is in multiples of 10^{-6}.

can be exploited in a progressive application (transmission, coding, and so on).

6. Conclusion and future works

We described an algorithm for optimizing two wavelet transforms for semi-regular meshes based on lifting scheme: the Butterfly- and the Loop-based transform. Our optimization consists in computing a prediction operator P that maximizes the sparsity of wavelet coefficients (by minimizing the L_1-norm) for improving the subsequent coding. This optimization is done at each level of resolution during the analysis. Experimental results show that our optimization algorithm improves significantly the coding of non-smooth models whatever the wavelet transform used. Visually, the adaptive Butterfly-based scheme also tends to further preserve the regions of high frequency details and the salient features. Finally, The Loop-based scheme creates low resolution meshes less distorted than the original transform, which is very interesting for progressive applications.

For the future, a thorough study of the statistics of the coefficients could be relevant to know how these lifting schemes can be further improved, and to develop new minimization criteria. In parallel, an in-depth evaluation of the performances of the algorithm in function of the input mesh could be also interesting. Another promising work could be the development of a region-based analysis for complex models. As proposed in the experimental section, it could be interesting to adapt the filters to several regions of the input, in function of the high frequency details (classical/adaptive transform for smooth/non smooth region for instance). Such an approach should improve the global quality of the encoded meshes.

Acknowledgments

The original (irregular mesh) VASE LION model is courtesy of *SenSable Technologies*. The original JOAN OF ARC model is courtesy of *I3S* Laboratory. The original BIMBA model is courtesy of *IMATI* and *INRIA*. The original RABBIT model is courtesy of *Cyberware*. All models are remeshed with Trireme [6]. We are particularly grateful to Igor Guskov for providing us with his executable (TRIREME). We also want to acknowledge the anonymous reviewers for their comments which permitted us to improve the quality of this paper.

Appendix A. Convexity of the function f

This section shows that the function f developed for improving the Butterfly-based lifting scheme is convex (see Section 3). We recall that the function f is given by

$$\begin{array}{rccl} f: & \mathbb{R}^{10} & \longmapsto & \mathbb{R} \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) = \|A\mathbf{x} - \mathbf{b}\|_1, \end{array}$$

with

$$\mathbf{x} = \left(\alpha_0^j, \alpha_0^j, \alpha_2^j, \alpha_2^j, \alpha_4^j, \alpha_4^j, \alpha_4^j, \alpha_4^j, \alpha_8^j, \alpha_8^j\right)^T.$$

By definition, the function f can be written as:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i=0}^{n_r-1} |A_i\mathbf{x} - b_i|, \\ &= \sum_{i=0}^{n_r-1} g_i(\mathbf{x}), \end{aligned}$$

where A_i are the rows of A, b_i are the values of $\mathbf{b}$, and n_r is the number of vertices of $\mathbf{V}_1^j$ on which the regular stencil is applied at this level of resolution. Considering $(\mathbf{x},\mathbf{y}) \in \mathbb{R}^{10}$, and $\lambda \in [0,1]$, we get

$$\begin{aligned} g_i(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) &= |A_i(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) - b_i|, \\ &= |\lambda A_i\mathbf{x} + (1-\lambda)A_i\mathbf{y} - b_i|, \\ &= |\lambda(A_i\mathbf{x} - b_i) + (1-\lambda)(A_i\mathbf{y} - b_i)|, \\ &\leq |\lambda(A_i\mathbf{x} - b_i)| + |(1-\lambda)(A_i\mathbf{y} - b_i)|. \end{aligned}$$

$\lambda \in [0,1]$, so λ and $(1-\lambda)$ are positive. We can deduce that

$$g_i(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda|A_i\mathbf{x} - b_i| + (1-\lambda)|A_i\mathbf{y} - b_i|,$$

and finally:

$$g_i(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda g_i(\mathbf{x}) + (1-\lambda)g_i(\mathbf{y}).$$

Thus, the function g_i is convex. Since the objective function f is a sum of g_i, f is also convex.

Appendix B. Computation of a_{ij} and b_i

This section gives details about the computation of the terms a_{ij} and b_i needed to find the weights $\boldsymbol{\rho}$ of the operator U_2 of the Loop-based lifting scheme (see Section 4.2).
The final goal is to solve $A\boldsymbol{\omega} = b$, where A is a symmetric 4×4 matrix

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix},$$

and b is a 4×1 matrix $b = [b_0\ b_1\ b_2\ b_3]^T$. a_{ij} and b_i are defined by the inner products $a_{ij} =< \phi_i, \phi_j >$ and $b_i =< \psi_l, \phi_i >$, with ϕ_i the i^{th} scaling function ($i = 0..4$) of the Loop transform, and ψ_l the lazy wavelet. To explain how computing a_{ij} and b_i, we give the examples of $b_1 =< \psi_l, \phi_1 >$ and $a_{11} =< \phi_1, \phi_1 >$. Figure B.15 shows the discrete scaling function ϕ_1 and the lazy wavelet, with the associated parameters. Note that the i^{th} scaling function ϕ_i is associated with the weight ω_i (Figures 10(c) and B.16). By superposing those two stencils and computing the discrete inner product, we can easily conclude that b_1 is equal to $\gamma_0\delta_0 + \alpha_1\delta_1 + \rho_0$, where

$$\begin{aligned} \alpha_i &= \rho_0 + \left(\rho_0 + 2\rho_1 \cos\frac{2\pi}{n_i}\right)^2, \\ \beta_i &= \frac{1}{1-\rho_0}(\alpha_i - \rho_0), \\ \gamma_i &= \frac{1-\alpha_i}{n_i}, \\ \delta_i &= \frac{1-\beta_i}{n_i}, \end{aligned}$$

n_i being the valence of the associated vertex. Similarly, the computation of a_{11} can be done by superposing the stencil of ϕ_1 on itself, and we obtain $\gamma_0^2 + \alpha_1^2 + \gamma_2^2 + \gamma_3^2 + (n_1 - 3)\gamma_6^2 + n_1(\rho_0^2 + \rho_1^2)$. Notice that, as Bertram in [19], we consider that the vertices of the scaling functions that are not on the Loop stencil (blue dots on figure B.16) are regular (valence 6). Therefore their value is equal to γ_6 on Figure B.15.

By following the same method on all the a_{ij} and b_i, we finally obtain for A

$$\begin{aligned} a_{00} &= \alpha_0^2 + \gamma_1^2 + \gamma_2^2 + \gamma_3^2 + (n_0 - 3)\gamma_6^2 + n_0(\rho_0^2 + \rho_1^2) \\ a_{01} &= a_{10} = \alpha_0\gamma_0 + \gamma_1\alpha_1 + \gamma_2^2 + \gamma_3^2 + \rho_0^2 + 4\rho_0\rho_1 \\ a_{02} &= a_{20} = \alpha_0\gamma_0 + \gamma_1^2 + \gamma_2\alpha_2 + \gamma_6^2 + \rho_0^2 + 4\rho_0\rho_1 \\ a_{03} &= a_{30} = \alpha_0\gamma_0 + \gamma_1^2 + \gamma_6^2 + \gamma_3\alpha_3 + \rho_0^2 + 4\rho_0\rho_1 \\ a_{11} &= \gamma_0^2 + \alpha_1^2 + \gamma_2^2 + \gamma_3^2 + (n_1 - 3)\gamma_6^2 + n_1(\rho_0^2 + \rho_1^2) \end{aligned}$$

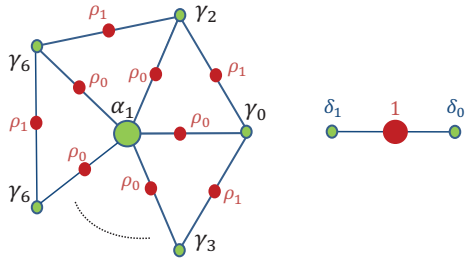


Figure B.15: Discrete scaling function ϕ_1 (left) and Lazy wavelet ψ_l (right) of the Loop-based scheme.

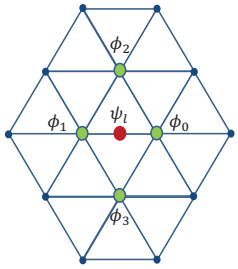


Figure B.16: Position of the four scaling functions and the lazy wavelet for the Loop-based prediction stencil.

$$\begin{aligned} a_{12} &= a_{21} = \gamma_0^2 + \alpha_1\gamma_1 + \gamma_2\alpha_2 + \gamma_6^2 + \rho_0^2 + 4\rho_0\rho_1 \\ a_{13} &= a_{31} = \gamma_0^2 + \alpha_1\gamma_1 + \gamma_6^2 + \gamma_3\alpha_3 + \rho_0^2 + 4\rho_0\rho_1 \\ a_{22} &= \gamma_0^2 + \gamma_1^2 + \alpha_2^2 + (n_2 - 2)\gamma_6^2 + n_2(\rho_0^2 + \rho_1^2) \\ a_{23} &= a_{32} = \gamma_0^2 + \gamma_1^2 + \rho_1^2 \\ a_{33} &= \gamma_0^2 + \gamma_1^2 + \alpha_3^2 + (n_3 - 2)\gamma_6^2 + n_3(\rho_0^2 + \rho_1^2). \end{aligned}$$

Similarly, we also obtain

$$\begin{aligned} b_0 &= (\alpha_0\delta_0 + \gamma_1\delta_1 + \rho_0) \\ b_1 &= (\gamma_0\delta_0 + \alpha_1\delta_1 + \rho_0) \\ b_2 &= b_3 = (\gamma_0\delta_0 + \gamma_1\delta_1 + \rho_1). \end{aligned}$$

References

[1] I. Daubechies, Orthonormal bases of compactly supported wavelets, Communications on Pure and Applied Mathematics 41 (1988) 909–996.

[2] S. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (7) (1989) 674–693.

[3] I. Guskov, W. Sweldens, P. Schröder, Multiresolution signal processing for meshes, in: A. Rockwood (Ed.), SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques, Addison Wesley Longman, 1999, pp. 325–334.

[4] S. Valette, R. Prost, A wavelet based compression scheme for triangle meshes : Wavemesh, IEEE Transactions on Visualization and Computer Graphics 10 (2) (2004) 123–133.

[5] I. Friedel, P. Schröder, A. Khodakovsky, Variational normal meshes, ACM Transactions on Graphics 23 (4) (2004) 1061–1073. doi:http://doi.acm.org/10.1145/1027411.1027418.
[6] I. Guskov, Manifold-based approach to semi-regular remeshing, Graphical Models 69 (1) (2007) 1–18. doi:http://dx.doi.org/10.1016/j.gmod.2006.05.001.
[7] A. Kammoun, F. Payan, M. Antonini, Adaptive semi-regular remeshing: A voronoi-based approach, in: Proceedings of IEEE international workshop on MultiMedia Signal Processing, Saint-Malo, France, 2010, pp. 350–355.
[8] M. Lounsbery, T. D. DeRose, J. Warren, Multiresolution analysis for surfaces of arbitrary topological type, ACM Transactions on Graphics 16 (1) (1997) 34–73.
[9] E. Catmull, J. Clark, Recursively generated B-spline surfaces on arbitrary topological meshes, Computer-aided Design 10 (1978) 350–355. doi:10.1016/0010-4485(78)90110-0.
[10] C. Loop, Smooth subdivision surfaces based on triangles, Master's thesis, University of Utah (1987).
[11] N. Dyn, D. Levin, J. Gregory, A butterfly subdivision scheme for surface interpolation with tension control, ACM Transactions on Graphics 9 (2) (1990) 160–169.
[12] D. L. Donoho, Interpolating wavelet transforms, Preprint, Department of Statistics, Stanford University (1992).
[13] P. Schröder, W. Sweldens, Spherical wavelets: efficiently representing functions on the sphere, in: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH '95, ACM, New York, NY, USA, 1995, pp. 161–172.
[14] S. Dahlke, W. Dahmen, I. Weinreich, E. Schmitt, Multiresolution analysis and wavelets on s2 and s3, Numerical functional analysis and optimization 16 (1,2) (1995) 19–41.
[15] W. Freeden, U. Windheuser, Spherical wavelet transform and its discretization, Advances in Computational Mathematics 5 (1) (1996) 51–94.
[16] W. Sweldens, The lifting scheme: A construction of second generation wavelets, SIAM Journal on Mathematical Analysis 29 (2) (1998) 511–546.
[17] J. Kovacevic, W. Sweldens, Wavelet families of increasing order in arbitrary dimensions, IEEE Transactions on Image Processing 9 (3) (2000) 480–496.
[18] A. Khodakovsky, P. Schröder, W. Sweldens, Progressive geometry compression, in: SIGGRAPH'00 Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co., 2000, pp. 271–278.
[19] M. Bertram, Biorthogonal loop-subdivision wavelets, Computing 72 (1-2) (2004) 29 – 39.
[20] D. Li, K. Qin, H. Sun, Unlifted loop subdivision wavelets, Pacific Conference on Computer Graphics and Applications 0 (2004) 25–33.
[21] M. Charina, J. Stöckler, Tight wavelet frames for subdivision, Journal of Computational and Applied Mathematics 221 (2) (2008) 293–301.
[22] P. Jingliang, K. Chang-Su, C.-C. J. Kuo, Technologies for 3D mesh compression: A survey, Journal of Visual Communication and Image Representation 16 (2005) 688–733.
[23] A. Khodakovsky, I. Guskov, Compression of normal meshes, in: Geometric Modeling for Scientific Visualization, Springer-Verlag, 2003, pp. 189–206.
[24] F. Payan, M. Antonini, An efficient bit allocation for compressing normal meshes with an error-driven quantization, Elsevier Computer Aided Geometry Design 22 (5) (2005) 466–486.
[25] F. Payan, M. Antonini, Mean square error approximation for wavelet-based semiregular mesh compression, IEEE Transactions on Visualization and Computer Graphics (TVCG) 12 (5) (2006) 649–657.
[26] J.-Y. Sim, C.-S. Kim, C. J. Kuo, S.-U. Lee, Normal mesh compression based on rate-distortion optimization, in: Proceedings of the IEEE Workshop on MultiMedia Signal Processing, 2002, pp. 13–16.
[27] S. Lavu, H. Choi, R. Baraniuk, Geometry compression of normal meshes using rate-distortion algorithms, in: Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing, Vol. 43, 2003, pp. 52–61.
[28] L. Denis, S. M. Satti, A. Munteanu, J. Cornelis, P. Schelkens, Scalable intraband and composite wavelet-based coding of semiregular meshes, IEEE Transactions on Multimedia 12 (8) (2010) 773–789.
[29] G. Piella, H. J. A. M. Heijmans, H. J. A. M. Heijmans, Adaptive lifting schemes with perfect reconstruction, IEEE Transactions on Signal Processing 50 (2001) 1620–1630.
[30] S. Mallat, A wavelet tour of signal processing, Academic Press, 1998.
[31] A. Kammoun, F. Payan, M. Antonini, Optimized butterfly-based lifting scheme for semi-regular meshes, in: Proceedings of IEEE International Conference in Image Processing (ICIP), Brussels, Belgium, 2011, pp. 1293–1296.
[32] P. Schröder, W. Sweldens, M. Cohen, T. DeRose, D. Salesin, Wavelets in computer graphics, SIGGRAPH Course notes (1996).
[33] D. Zorin, P. Schröder, W. Sweldens, Interpolating subdivision for meshes with arbitrary topology, in: Proceedings of SIGGRAPH 96, 1996, pp. 189–192.
[34] D. Donoho, Y. Tsaig, Fast solution of l_1-norm minimization problems when the solution may be sparse, IEEE Transactions on Information Theory 54 (11) (2008) 4789–4812.
[35] J. C. Lagarias, J. C. Lagarias, J. A. Reeds, J. A. Reeds, M. H. Wright, M. H. Wright, P. E. Wright, P. E. Wright, Convergence properties of the nelder-mead simplex algorithm in low dimensions, SIAM Journal of Optimization 9 (1996) 112–147.
[36] C. Touma, C. Gotsman, Triangle mesh compression, Graphics Interface'98 (1998) 26–34.
[37] N. Aspert, D. Santa-Cruz, T. Ebrahimi, Mesh: Measuring errors between surfaces using the hausdorff distance, in: IEEE International Conference in Multimedia and Expo (ICME), Vol. 1, 2002, pp. 705–708.